Joint Speech Recognition and Speaker Diarization via Sequence Transduction

Laurent El Shafey, Hagen Soltau, Izhak Shafran

Google

shafey@google.com, soltau@google.com, izhak@google.com

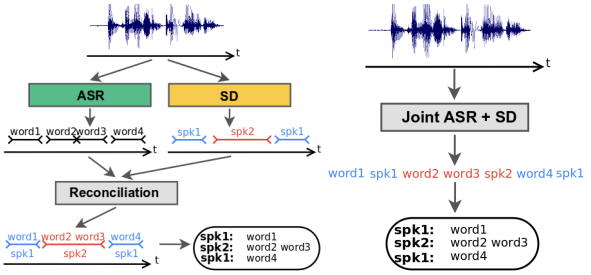
Abstract

Speech applications dealing with conversations require not only recognizing the spoken words, but also determining who spoke when. The task of assigning words to speakers is typically addressed by merging the outputs of two separate systems, namely, an automatic speech recognition (ASR) system and a speaker diarization (SD) system. The two systems are trained independently with different objective functions. Often the SD systems operate directly on the acoustics and are not constrained to respect word boundaries and this deficiency is overcome in an *ad hoc* manner. Motivated by recent advances in sequence to sequence learning, we propose a novel approach to tackle the two tasks by a joint ASR and SD system using a recurrent neural network transducer. Our approach utilizes both linguistic and acoustic cues to infer speaker roles, as opposed to typical SD systems, which only use acoustic cues. We evaluated the performance of our approach on a large corpus of medical conversations between physicians and patients. Compared to a competitive conventional baseline, our approach improves word-level diarization error rate from 15.8% to 2.2%.

1. Introduction

In the last few decades, speech and language technology has advanced significantly, leading to a profound change in the way people interact with machines and low cost devices. For instance, with the rapid growth of smart speakers, automatic speech recognition (ASR) systems are now commonly used by millions of users. Even with these remarkable advances, machines have difficulties understanding natural conversations with multiple speakers such as in broadcast interviews, meetings, telephone calls, videos or medical recordings. One of the first steps in understanding natural conversations is to recognize the words spoken and their speakers. As illustrated in Figure 1a, this is typically performed in multiple steps that include (1) transcribing the words using an ASR system, (2) predicting "who spoke when" using a speaker diarization (SD) system, and, finally, (3) reconciling the output of those two systems.

More formally, speaker diarization consists of partitioning an input audio stream into time-bounded segments before annotating each of those segments with a speaker label. Many different SD systems have been proposed in the literature [1, 2] and they often rely on the following pattern: (a) run a voice activity detector and segment the input audio into speech segments, (b) extract features from each segment to generate a speaker embedding and (c) cluster the resulting speaker embeddings. While early work relied on handcrafted audio features for speaker embedding [3, 4], recent efforts have been successful in learning better representations automatically using i-vectors [5] in feed-forward neural networks [6] or recurrent neural networks (RNN) [7]. The embeddings have been further improved by explicitly maximizing speaker classification accuracy [8, 9]. Triplet loss was proposed as an alternative cost function though



(a) *Baseline* (b) *Joint ASR+SD*

Figure 1: *Comparison of the conventional speech recognition and speaker diarization system (Figure 1a) with the proposed approach (Figure 1b), where the task consists of generating a speaker-decorated transcript from raw audio.*

the training requirements are cumbersome [10]. In one variant, the clustering step has been successfully replaced with a supervised approach [11]. One commonality with most of the previous work, is that they rely solely on acoustic information to assign speaker labels to audio segments.

However, in many speech applications, subjects in a conversation play very specific roles, which is reflected in their language use. For example, in a clinical visit, the physicians ask all the questions about the symptoms that the patient may be experiencing. Similarly, the patients are likely to be the ones seeking clarifications about treatments. Speaker labels could possibly be inferred from the transcript directly. Only a few approaches have utilized such linguistic cues for speaker diarization. In one approach, linguistic cues were used to associate speaker labels to segments generated by a conventional SD system [12]. More recently, a gated recurrent unit-based sequence to sequence model was employed to detect speaker changes [13]. The changes were predicted over 32 word sliding windows and the predictions from overlapping windows were resolved using a voting mechanism, which was then followed by a traditional clustering step.

We propose a novel method to perform automatic speech recognition and speaker diarization in a joint manner, as illustrated in Figure 1b. Our approach utilizes both acoustic and linguistic cues, and is, hence, designed to perform well in scenarios where the speakers involved have well-defined roles. Specifically, our main contribution consists of defining the joint ASR and SD task as a sequence transduction problem and implementing the solution using a recurrent neural network transducer (RNN-T) (Section 2). We train and evaluate this system on a large corpus of clinical conversations, which is a very good fit for such a joint acoustic and linguistic SD system. The experimental results highlight significant improvements compared to a strong baseline using a conventional SD system (Section 3). Finally, we summarize our findings and provide future lines of research (Section 4).

hello dr jekyll <spk:pt> hello mr hyde what brings you here today <spk:dr> I am struggling again with my bipolar disorder <spk:pt>

Figure 2: *Example of an output sequence for our joint ASR and SD RNN-T system. The corresponding input would be the raw audio signal. Speaker turns are displayed in different colors.*

2. Diarization via Sequence Transduction

2.1. Problem Formulation and Proposed Solution

Many machine learning tasks can be expressed as mapping an input sequence into an output sequence. Specifically, speech recognition can be defined as a transformation that outputs a sequence of words from an audio signal. RNNs are popular models that have been used to model such sequential data. In speech recognition, they are often used in a hybrid setting, where alignments are precomputed via the Viterbi algorithm using an existing model [14]. As an alternative, Connectionist temporal classification (CTC) does not rely on pre-computed alignments and maximizes the likelihood via executing the forward-backward algorithm at every step [15]. To accommodate different sequence length between input and output, a blank symbol is introduced and the loss is calculated by marginalizing over all possible alignments. The main shortcoming of CTC is that output dependencies are not modeled. RNN-T models can be seen as an extension of CTC that addresses this shortcoming and adds a language model component [16]. Initially, the scores from the acoustic and language model components were simply multiplied. This was later improved up on by using a general feed forward layer [17] and the resulting model proved to be very successful in various sequence modeling tasks.

Compared to conventional systems where acoustic and language models are trained separately, RNN-T models have recently been successfully applied to speech recognition using end-to-end training, leading to comparable or even higher accuracy on a diverse set of acoustic conditions [17, 18].

The two key insights we exploit in our work are: (a) RNN-Ts can output richer set of targets symbols such as speaker role and punctuation, since they allow prediction of symbols without the need to explicitly attach observations to them, and (b) they can seamlessly integrate acoustic and linguistic information.

As a proof of concept, this article focuses on enriching the traditional speech recognition units with speaker roles, as illustrated in Figure 1. The closest approach we could find in the literature is [13], which makes use of a sequence to sequence model, but only for SD and not ASR. For this task, we augment the output symbol set with new speaker role tokens. In the case of medical conversations between a physician and a patient, those additional tokens would e.g. be <spk:dr> and <spk:pt>. For a given audio input snippet consisting of a speech conversation between a physician and a patient, our RNN-T model predicts a joint ASR and SD output sequence, as illustrated by an example in Figure 2.

2.2. Recurrent Neural Network Transducer

Let $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$ be an input sequence of acoustic frames[1], where T is the number of frames in the sequence.

[1]As a naming convention, we employ uppercase bold symbols to denote variables that represent sequences over time, and corresponding lowercase bold symbols for elements within such a sequence at given time steps.

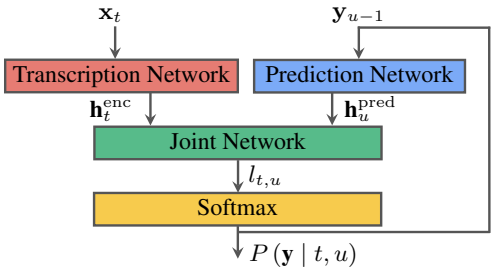


Figure 3: *Architecture of the RNN-T-based model.*

Typically, $\mathbf{x}_t \in \mathbb{R}^d$ are log-mel filterbank energies. Let $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_U)$ be the corresponding output sequence of symbols (including speaker roles) over the RNN-T output space $\mathcal{Y}$, and $\mathcal{Y}^*$ be the set of all possible sequence over $\mathcal{Y}$. To handle different alignments between input and output sequences, we define an extended output space $\bar{\mathcal{Y}} = \mathcal{Y} \cup \{\varnothing\}$, where $\varnothing$ denotes the blank output, as well as the corresponding set $\bar{\mathcal{Y}}^*$ of all possible sequences over $\bar{\mathcal{Y}}$. An element $\mathbf{A} \in \bar{\mathcal{Y}}^*$ is referred to as an alignment, because the location of the blank symbols defines a mapping between the input and output symbols. For instance, the sequence $\mathbf{A} = (\mathbf{y}_1, \varnothing, \mathbf{y}_2, \varnothing, \varnothing, \mathbf{y}_3) \in \bar{\mathcal{Y}}^*$ is then equivalent to $(\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3) \in \mathcal{Y}^*$. Given the input sequence of frames $\mathbf{X}$, the RNN-T model [16] defines a conditional probability $P(\mathbf{Y} \in \mathcal{Y}^* \mid \mathbf{X})$ by marginalizing over the possible alignments

$$P(\mathbf{Y} \in \mathcal{Y}^* \mid \mathbf{X}) = \sum_{\mathbf{A} \in \mathcal{B}^{-1}(\mathbf{Y})} P(\mathbf{A} \mid \mathbf{X}), \tag{1}$$

where $\mathcal{B}$ is the function that removes the blank symbols from a given alignment in $\bar{\mathcal{Y}}^*$.

An RNN-T models this conditional probability using three different networks. (1) A *transcription network* (commonly called *encoder* in the sequence to sequence literature) maps the acoustic frames $\{\mathbf{x}_\tau\}_{1 \le \tau \le t}$ into a higher level representation $\mathbf{h}_t^{\text{enc}} = f^{\text{enc}}\left(\{\mathbf{x}_\tau\}_{1 \le \tau \le t}\right)$. (2) A *prediction network* makes predictions based on the previous non-blank symbol such that $\mathbf{h}_u^{\text{pred}} = f^{\text{pred}}\left(\{\mathbf{y}_v\}_{1 \le v \le u-1}\right)$. Finally, (3) a *joint network* combines the output of the previous two networks to produce the logits $l_{t,u} = f^{\text{joint}}\left(\mathbf{h}_t^{\text{enc}}, \mathbf{h}_u^{\text{pred}}\right)$. The output of the joint network is then fed into a softmax layer to define a probability distribution. The model is optimized by maximizing the log-likelihood of $P(\mathbf{Y} \in \mathcal{Y}^* \mid \mathbf{X})$.

The forward-back algorithm is used to calculate $P(\mathbf{Y} \in \mathcal{Y}^* \mid \mathbf{X})$ efficiently via dynamic programming. Training the RNN-T model on accelerators like graphical processing units (GPU) or tensor processing units (TPU [19]) is non-trivial as computation of the loss function requires running the forward-backward algorithm. This issue was addressed recently in a TPU friendly implementation of the forward-backward algorithm, which recasts the problem as a sequence of matrix multiplications [20]. We took advantage of an efficient implementation of the RNN-T loss in TensorFlow that allowed quick iterations of model development [21].

2.3. Model Implementation

For training purposes, we split long conversations into audio segments of maximum 15s that may contain multiple speakers. The corresponding output targets are speaker role decorated transcripts as previously depicted on Figure 2. We then

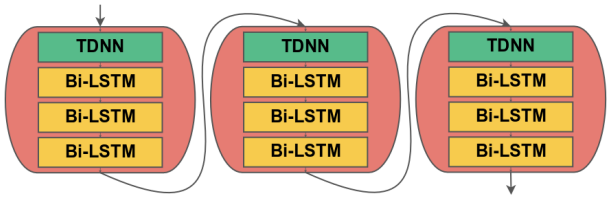


Figure 4: *Transcription network (encoder) architecture*

extract acoustic frames, which are 80-dimensional logmel filterbank energies ($d = 80$).

While sequence to sequence models often make use of graphemes as units, we argue that longer units are more appropriate for speech recognition. For example, if training data is abundant, entire words can be modeled directly in an LVCSR system [22]. In this work, we choose a middle ground, and use morphemes as units that were obtained by a data driven approach [23]. Longer units have the advantage that we can reduce the time resolution for the output sequence, making both training and inference more efficient. We do this by employing a hierarchy of time delay neural network (TDNN) layers that reduces the time resolution from 10ms to 80ms [24]. The architecture is very similar to the encoder used for CTC word models where the time resolution was reduced to 120ms and this decimation improved both inference speed and word error rate [25].

Specifically, our encoder consists of three identical blocks made of four layers (see Figure 4): (a) a $1D$ temporal convolutional layer with 512 filters, a kernel size of 5 and a max-pooling operator of size 2 followed by (b) three bi-directional long short-term memory (LSTM [26]) layers with 512 units. The prediction network consists of a word embedding layer that maps our morpheme vocabulary of about 4K units to a 512 dimension vector space followed by a uni-directional LSTM with 1024 units and a fully connected layer with 512 units.

Training is performed using the stochastic gradient-based Adam optimizer [27]. We train the model on a set of 128 TPUs and it converges in less than two days.

3. Experiments

3.1. Corpus

We experimented on a large corpus of about 100K ($\approx$ 15K hours) manually transcribed audio recordings of clinical conversations between physicians and patients, where each conversation is about 10 minutes long on the average. The transcription breaks up a conversation into speaker turns and in each turn identifies the speaker role label and the sequence of words spoken. This long-form transcription makes the corpus well-suited for training our proposed model. Each conversation typically contains two different speaker roles, respectively `<spk:dr>` and `<spk:pt>`. For the relatively few cases where other speakers are involved, we map them to the closest speaker role (e.g., nurses to `<spk:dr>`, family caregivers to `<spk:pt>`). The acoustic quality of the recordings varied significantly due to several factors – the distance of the speakers to the microphone, recording devices, the audio encoder (mp3 or wav) and the sample rate (8kHz or 16kHz). We partitioned our corpus into three sets, holding out 508 and 404 conversations for development and evaluation sets respectively, and the rest for training. There are no overlaps of physicians between the three sets. The patient overlap is uncertain since their identities are unavailable. The training data is split into segments of less than 15sec, as mentioned earlier, and results in an average of about 4 speaker turns per segment.

3.2. Evaluation Metric

The common approach to evaluate conventional diarization systems is to rely on the diarization error rate (DER) metric, which compares reference speaker-labeled segments with SD predictions in the time domain. In contrast, the use of a joint ASR and SD system directly assigns speaker roles to the recognized words, and hence makes it unnecessary to depend on the time boundaries for aligning words with the speaker roles.

Motivated by the speaker attributed task proposed in the NIST RT-03F evaluation plan [28] (Section 5.2.5), we utilize a metric suitable to assess such end-to-end joint ASR and SD systems, by measuring the percentage of words in the transcript decorated with the right speaker tag. Specifically, we define the Word Diarization Error Rate (WDER) as:

$$\text{WDER} = \frac{S_{\text{IS}} + C_{\text{IS}}}{S + C} \tag{2}$$

where,

1. S_{IS} is the number of ASR Substitutions with Incorrect Speaker tokens,
2. C_{IS} is the number of Correct ASR words with Incorrect Speaker tokens,
3. S is the number of ASR substitutions,
4. C is the number of Correct ASR words.

Note that this WDER metric must be used in combination with the ASR Word Error Rate (WER) to account for deletions and insertions since the speaker labels associated with them cannot be mapped to reference without ambiguity. In our opinion, this word-level metric reflects the performance in an actual application better than the time-level metric.

3.3. Baseline

Since our proposed joint model is based on RNN-T, we compare its performance with a baseline system that also uses an equivalent RNN-T model but only for ASR. The architecture of the ASR system is same as described in Section 2.3, except that the speaker roles are removed from the transcript. Based on our past experience with conventional SD system, we built a strong baseline system consisting of the following five stages:

(a) **Speech detection and segmentation**: This stage consists of an LSTM-based speech detector whose threshold is kept low to minimize deletion of speech segments [29].

(b) **Speaker embedding**: The speaker embeddings are computed using a sliding window of 1sec with a stride of 100ms. An acoustic feature sequence is extracted from the 1sec window and fed into an LSTM model. The last hidden state of the LSTM is extracted as the embedding for the 1sec window [30]. The embeddings are trained to maximize classification of speakers in the training set [8, 9].

(c) **Speaker change detection**: Cosine distance is computed between adjacent embedding vectors in the speech segments. When the distance is higher than a threshold (optimized on the development set), the transition is marked as a speaker change. Thus, speech segments are broken into single speaker segments.

(d) **Speaker clustering**: The single speaker segments are clustered so that the speaker labels can be applied consistently across all segments from the same speaker. Among various choices of clustering algorithms, we found k-means to be most effective, with $k = 2$.

(e) **Reconciling the ASR and SD output**: The SD system provides speaker turns with time boundaries and these labels are

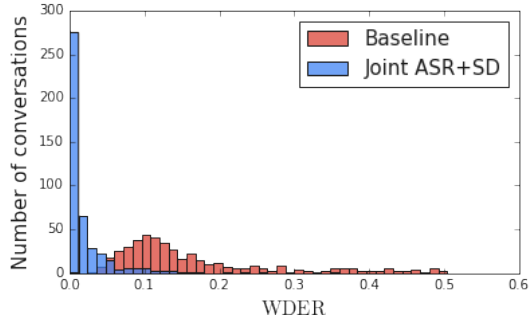


Figure 5: *Distribution of the* WDER *on a per conversation basis for the baseline and the proposed system.*

mapped to recognized words using the associated word boundaries from the ASR system. When the speaker turn boundary fall in the middle of a word, we assign the word to the speaker with the largest overlap with the word. The baseline predicts generic speaker tags such as `<spk:0>` and `<spk:1>`.

For evaluation purposes, we map the generic labels back to speaker roles on a per conversation basis. The mappings are picked to minimize the diarization errors, hence, giving an advantage to the baseline over the proposed system. For training speaker embeddings, we augmented our training data with the VoxCeleb2 dataset [31] since it improved the performance over using only our corpus. Note, the VoxCeleb2 data does not contain long-form transcription or speaker role labels in medical domain and hence cannot be used for training the integrated RNN-T models. Thus, the baseline used more data than our RNN-T model.

3.4. Results and Analysis

In Table 1, we compare the performance of our system with the strong baseline described above. We observe a substantial improvement in WDER, which drops from 15.8% to 2.2%, a relative improvement of about 86% over the baseline. This gain in WDER comes at a small cost in ASR performance with about 0.6% degradation in WER.

In a conventional system, the WDER is affected by several factors such as errors in ASR generated word boundaries, errors in SD generated speaker turn boundaries as well as errors in the ad hoc reconciliation step. In contrast, our proposed system clearly benefits from the lack of such intermediate steps.

Examining the distribution of errors across conversations, as shown in Figure 5, we find that the performance of our proposed system is much more consistent with most of the conversations under WDER of 4%, while the baseline system has a wide spread around a mode of 11%. On manual inspection of the outliers of our system (above 15%), interestingly, the speaker change detections rarely fail, but, once the model

	Baseline	Joint ASR+SD
WDER	15.8%	**2.2%**
WER	**18.7%**	19.3%
D/I/S	7.2%/2.1%/9.4%	6.8%/2.8%/9.7%

Table 1: *Word Diarization Error Rate (*WDER*), Word Error Rate (*WER*) and its decomposition in Deletion / Insertion / Substitution errors (*D/I/S*) on the evaluation set.*

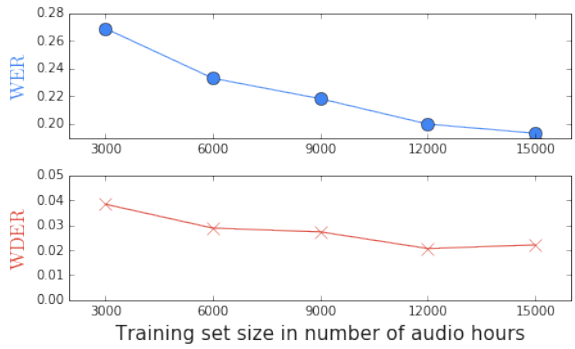


Figure 6: *Impact of amount of training data on the performance of the proposed system on the evaluation set.*

wrongly assigns speaker roles, they stay flipped for a large segment of the conversation. We suspect this is due to the way inference is performed on chunks of conversation in batches and not on the entire conversation.

Joint models are notorious for being susceptible to data sparsity, so we investigate the impact of training data size on our model performance. In the Figure 6, we plot the performance with respect to training data size ranging from $3,000$ to $15,000$ hours of audio. For ease of experimentation, we didn't tune the model size. We observe that more training data seems to be more helpful to improve WER rather than WDER.

4. Conclusions And Future Work

We introduced a novel joint ASR and SD system, which relies on the sequence to sequence paradigm and is implemented using an RNN-T model. We demonstrated the performance of our approach by evaluating it on a large corpus of clinical conversations between physicians and patients. Compared to a conventional baseline, we observed a significant relative improvement of 86% in the word-level diarization error rate, without significant degradation in the word error rate.

This system is particularly well-suited for applications where there are limited number of speakers in a conversation and speaker roles fall into well-defined categories. Note that the focus of our system is labeling speaker roles rather than identities, and as such it is critical to have matched speaker role labeled training data. Unfortunately, to the best of our knowledge, there is no public corpus where we could have evaluated our model for the benefit of the larger community. One other question that we were unable to address in this work is how much of the performance gain was from the lexical cues. This is complicated by the fact that lexical cues need to be inferred on spoken words and not transcribed ones.

In the future, we would like to evaluate our approach on other applications with clear speaker roles. Longer term, we are interested in providing truly rich transcripts of conversations. For example, we are currently conducting experiments to include punctuation and capitalization that look promising and we believe that the approach could be highly beneficial for exploiting non-verbal cues such as emotions.

5. Acknowledgements

We are grateful to Rick Rose and Olivier Siohan for many discussions and help with the baseline system, the WDER metric and its implementation, and to Gang Li for help with improving the speaker embedding for the baseline system.

6. References

[1] S. E. Tranter and D. A. Reynolds, "An overview of automatic speaker diarization systems," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1557–1565, 2006.

[2] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, "Speaker diarization: A review of recent research," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 2, pp. 356–370, 2012.

[3] J. Ajmera and C. Wooters, "A robust speaker clustering algorithm," in *IEEE Workshop on Automatic Speech Recognition and Understanding*. IEEE, 2003, pp. 411–416.

[4] C. Barras, X. Zhu, S. Meignier, and J.-L. Gauvain, "Multistage speaker diarization of broadcast news," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1505–1512, 2006.

[5] G. Sell and D. Garcia-Romero, "Speaker diarization with PLDA i-vector scoring and unsupervised calibration," in *IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2014, pp. 413–417.

[6] D. Garcia-Romero, D. Snyder, G. Sell, D. Povey, and A. McCree, "Speaker diarization using deep neural network embeddings," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4930–4934.

[7] Q. Wang, C. Downey, L. Wan, P. A. Mansfield, and I. L. Moreno, "Speaker diarization with LSTM," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5239–5243.

[8] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[9] G. Sell, D. Snyder, A. McCree, D. Garcia-Romero, J. Villalba, M. Maciejewski, V. Manohar, N. Dehak, D. Povey, S. Watanabe, and S. Khudanpur, "Diarization is hard: Some experiences and lessons learned for the JHU team in the inaugural DIHARD challenge," in *Interspeech*. ISCA, 2018, pp. 2808–2812.

[10] H. Bredin, "Tristounet: Triplet loss for speaker turn embedding," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 5430–5434.

[11] A. Zhang, Q. Wang, Z. Zhu, J. Paisley, and C. Wang, "Fully supervised speaker diarization," *arXiv preprint arXiv:1810.04719*, 2018.

[12] L. Canseco-Rodriguez, L. Lamel, and J.-L. Gauvain, "Speaker diarization from speech transcripts," in *Interspeech / International Conference on Spoken Language Processing (ICSLP)*, vol. 4. IEEE, 2004, pp. 3–7.

[13] T. J. Park and P. G. Georgiou, "Multimodal speaker segmentation and diarization using lexical and acoustic cues via sequence to sequence neural networks," in *International Speech Communication Association*, 2018, pp. 1373–1377.

[14] T. Robinson, M. Hochberg, and S. Renals, "The use of recurrent neural networks in continuous speech recognition," in *Automatic speech and speaker recognition*. Springer, 1996, pp. 233–258.

[15] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *International Conference on Machine Learning*, ser. ACM International Conference Proceeding Series, vol. 148. ACM, 2006, pp. 369–376.

[16] A. Graves, "Sequence transduction with recurrent neural networks," *CoRR*, 2012.

[17] A. Graves, A. Mohamed, and G. E. Hinton, "Speech recognition with deep recurrent neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2013, pp. 6645–6649.

[18] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang, Q. Liang, D. Bhatia, Y. Shangguan, B. Li, G. Pundak, K. C. Sim, T. Bagby, S. Chang, K. Rao, and A. Gruenstein, "Streaming end-to-end speech recognition for mobile devices," *arXiv preprint arXiv:1811.06621*, 2018.

[19] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *ACM/IEEE Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2017, pp. 1–12.

[20] K. C. Sim, A. Narayanan, T. Bagby, T. N. Sainath, and M. Bacchiani, "Improving the efficiency of forward-backward algorithm using batched computation in tensorflow," in *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017, pp. 258–264.

[21] T. Bagby and K. Rao, "Efficient implementation of recurrent neural network transducer in tensorflow," in *IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018.

[22] H. Soltau, H. Liao, and H. Sak, "Neural speech recognizer: Acoustic-to-word LSTM model for large vocabulary speech recognition," in *Interspeech*. ISCA, 2017.

[23] S. Virpioja, P. Smit, S.-A. Grönroos, and M. Kurimo, "Morfessor 2.0: Python implementation and extensions for morfessor baseline," Aalto University, Tech. Rep., 2013.

[24] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *Backpropagation: Theory, Architectures and Applications*, pp. 35–61, 1995.

[25] H. Soltau, H. Liao, and H. Sak, "Reducing the computational complexity for whole word models," in *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[28] Anonymous, "The Rich Transcription Fall 2003 (RT-03F) Evaluation Plan," NIST, Tech. Rep., 2003.

[29] R. Zazo, T. N. Sainath, G. Simko, and C. Parada, "Feature learning with raw-waveform cldnns for voice activity detection," in *Interspeech*. ISCA, 2016.

[30] G. Heigold, I. Moreno, S. Bengio, and N. M. Shazeer, "End-to-end text-dependent speaker verification," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[31] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *Interspeech*. ISCA, 2018.